



## My purpose

To build a health system  
that takes everyone along

Corporate Bangalore | Bengaluru | Ranchi

## My programme

Master of Public Health

Admissions Open

Clear



main.c



Share



Run

Output

```
45     second = (struct Node*)malloc(sizeof(struct Node));
46     third = (struct Node*)malloc(sizeof(struct Node));
47
48     head->data = 10;
49     head->next = first;
50
51     first->data = 20;
52     first->next = second;
53
54     second->data = 30;
55     second->next = third;
56
57     third->data = 40;
58     third->next = NULL;
59
60     printf("Original List:\n");
61     display(head);
62
63     deleteAtEnd(&head);
64
65     printf("After deleting last node:\n");
66     display(head);
67
68     return 0;
69 }
```

Original List:

10 -> 20 -> 30 -> 40 -> NULL

After deleting last node:

10 -> 20 -> 30 -> NULL

==== Code Execution Successful ===



main.c



Share

Run

Output

Clear

```
27     free(temp->next);
28     temp->next = NULL;
29 }
30
31 - void display(struct Node *head) {
32     struct Node *temp = head;
33     while (temp != NULL) {
34         printf("%d -> ", temp->data);
35         temp = temp->next;
36     }
37     printf("NULL\n");
38 }
39
40 - int main() {
41     struct Node *head, *first, *second, *third;
42
43     head = (struct Node*)malloc(sizeof(struct Node));
44     first = (struct Node*)malloc(sizeof(struct Node));
45     second = (struct Node*)malloc(sizeof(struct Node));
46     third = (struct Node*)malloc(sizeof(struct Node));
47
48     head->data = 10;
49     head->next = first;
50
51     first->data = 20;
52     first->next = second;
```

## ^ Original List:

10 -&gt; 20 -&gt; 30 -&gt; 40 -&gt; NULL

After deleting last node:

10 -&gt; 20 -&gt; 30 -&gt; NULL

== Code Execution Successful ==



main.c



Run

Output

Clear

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 struct Node {
5     int data;
6     struct Node *next;
7 };
8
9 void deleteAtEnd(struct Node **head) {
10    if (*head == NULL) {
11        printf("List is empty\n");
12        return;
13    }
14
15    if ((*head)->next == NULL) {
16        free(*head);
17        *head = NULL;
18        return;
19    }
20
21    struct Node *temp = *head;
22
23    while (temp->next->next != NULL) {
24        temp = temp->next;
25    }
26}
```

Original List:

10 -&gt; 20 -&gt; 30 -&gt; 40 -&gt; NULL

After deleting last node:

10 -&gt; 20 -&gt; 30 -&gt; NULL

== Code Execution Successful ==



main.c



Share

Run

Output

Clear

```
25      }
26      printf("NULL\n");
27  }
28
29- int main() {
30     struct Node *head, *first, *second, *third;
31     head = (struct Node*)malloc(sizeof(struct Node));
32     first = (struct Node*)malloc(sizeof(struct Node));
33     second = (struct Node*)malloc(sizeof(struct Node));
34     third = (struct Node*)malloc(sizeof(struct Node));
35     head->data = 10;
36     head->next = first;
37     first->data = 20;
38     first->next = second;
39     second->data = 30;
40     second->next = third;
41     third->data = 40;
42     third->next = NULL;
43     printf("Original List:\n");
44     display(head);
45     deleteAtStart(&head);
46     printf("After deleting first node:\n");
47     display(head);
48
49     return 0;
50 }
```

Original List:

10 -&gt; 20 -&gt; 30 -&gt; 40 -&gt; NULL

After deleting first node:

20 -&gt; 30 -&gt; 40 -&gt; NULL

==== Code Execution Successful ===

Get right to the  
right conversations.



with AI features that help you  
build advocacy in your accounts.



Learn more

main.c



Run

Output

Clear

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 struct Node {
5     int data;
6     struct Node *next;
7 };
8
9 void deleteAtStart(struct Node **head) {
10    if (*head == NULL) {
11        printf("List is empty\n");
12        return;
13    }
14
15    struct Node *temp = *head;
16    *head = (*head)->next;
17    free(temp);
18 }
19
20 void display(struct Node *head) {
21    struct Node *temp = head;
22    while (temp != NULL) {
23        printf("%d -> ", temp->data);
24        temp = temp->next;
25    }
26    printf("NULL\n");
```

Original List:  
10 -> 20 -> 30 -> 40 -> NULL  
After deleting first node:  
20 -> 30 -> 40 -> NULL

== Code Execution Successful ==



main.c



Share



Run

Output

```
51     second = (struct Node*)malloc(sizeof(struct Node));
52     third = (struct Node*)malloc(sizeof(struct Node));
53
54     head->data = 10;
55     head->next = first;
56
57     first->data = 20;
58     first->next = second;
59
60     second->data = 30;
61     second->next = third;
62
63     third->data = 40;
64     third->next = NULL;
65
66     printf("Original List:\n");
67     display(head);
68
69     int position = 3;
70     deleteAtPosition(&head, position);
71
72     printf("After deleting node at position %d:\n", position);
73     display(head);
74
75     return 0;
76 }
```

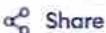
Original List:

10 -&gt; 20 -&gt; 30 -&gt; 40 -&gt; NULL

After deleting node at position 3:

10 -&gt; 20 -&gt; 40 -&gt; NULL

==== Code Execution Successful ===



main.c

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 struct Node {
5     int data;
6     struct Node *next;
7 };
8
9 void deleteAtPosition(struct Node **head, int position) {
10    if (*head == NULL) {
11        printf("List is empty\n");
12        return;
13    }
14
15    struct Node *temp = *head;
16
17    if (position == 1) {
18        *head = temp->next;
19        free(temp);
20        return;
21    }
22
23    for (int i = 1; temp != NULL && i < position - 1; i++) {
24        temp = temp->next;
25    }
26}
```

Output

Original List:

```
10 -> 20 -> 30 -> 40 -> NULL
After deleting node at position 3:
10 -> 20 -> 40 -> NULL
```

```
==== Code Execution Successful ===
```